Détection de structures communautaires dans des réseaux

SENE, CÔME, PRALON

20 Octobre 2022

Définitions et hypothèses

Représentation

$$\textit{Soit } G = (V, E), \text{ un graphe simple tels que}$$

$V = \{v_1, \cdots, v_p\}$ l'ensemble des nœuds
$E \subset \{(v_i, v_j)_{i,j \in \{1,\cdots,p\}} | i \neq j\} = V \times V$ l'ensemble des arêtes du graphe

Densité d'un graphe

On appel densité d'un graphe la valeur

$$D_G = \frac{|E|}{\frac{p^2 - p}{2}}$$

Correspond à la fréquence d'arêtes dans le graphe, elle rend compte de la connexion entre les nœuds

Définitions et hypothèses

Degré d'un noeud

On appel degré d'un nœud i la valeur

$$d_i = |\{(v_i, v_j) \in E | j \in \{1, \cdots, p\}\}|$$

et correspond au nombre de voisin du noeud i.

Modèle nul

On appelle modèle nul d'un graph G, le graph G^* dont les $|E| = m$ arêtes ont été distribuées aléatoirement entre les nœuds de G

Le modèle nul joue un modèle de référence pour lequel il n'existe aucune structure communautaire dans le réseau.

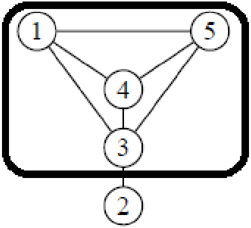


Figure – Détection d'une bonne partition en 2 classes

Modularité

Modularité

Soit $(C_1, \cdots, C_K)$ une partition de V. On définit la modularité Q de la partition comme

$$Q(C_1, \cdots, C_K) = \frac{1}{2m} \sum_{k=1}^{K} \sum_{(v_i, v_j) \in C_k} (1_{(v_i, v_j) \in E} - P_{i,j})$$

avec 2m $= \sum_{i=1}^{p} d_i$ et $P_{i,j} = \frac{d_i d_j}{2m}$
la probabilité que v_i et v_j soient connectés sous le modèle nul

Méthode de Louvain

Etape 1 de l'algorithme

Il s'agit d'un algorithme itératif approchant la partition de modularité maximale en complexité temporelle $\mathcal{O}(nlog(n))$

- Choix d'un parcours aléatoire
- Affécter chaque noeud à sa propre communauté
- Agrégation de i à la communauté du voisin j si

$$\Delta Q = Q(C_1, \cdots, C_i, \cdots, C_j, \cdots, C_p) - Q(C_1, \cdots, C_{i,j}, \cdots, C_p) > 0$$

  et plus grand que les autres voisins de i
- itération sur le chemin

Méthode de Louvain

Etape 2 de l'algorithme

- Agrégation des noeuds en communauté
- On obtient un nouveau réseau, pondéré
- Itération de l'étape 1
- Arrêt lorsque $\Delta Q = 0$

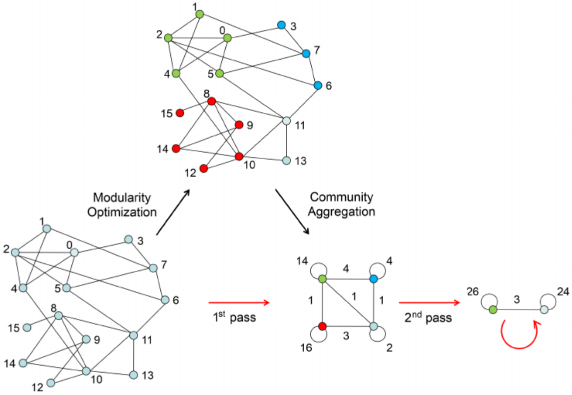


Figure – Exemple d'application de la méthode de Louvain

Expériences numériques

Environnement et librairies utilisées

- Langage : Python 3.9
- Environnement : Spyder
- Librairies : *igraph*, *matplotlib*, *random*

Graphes utilisés

- Graphe du club de karaté de Zachary
- Graphe généré aléatoirement avec le modèle de *Ernos Rényi*

La fonction *DetectCommunities*

Son objectif

Détecter les communautés d'un graphe donné

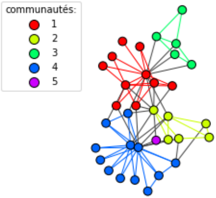
Ses arguments

Un objet de type *igraph.Graph* (graphe donné par l'utilisateur)

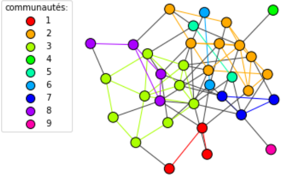
Ce qu'elle retourne

Un graphe avec ses communautés mise en évidence par des couleurs

Expériences numériques



(a) communautés détectés pour le graphe "Zachary"



(b) communautés détectés pour le graphe généré aléatoirement

1 WikiStat. An introduction to network inference and mining, *Article*
http://www.nathalievialaneix.eu/doc/pdf/wikistat-network_compiled.pdf

2 PNAS. Modularity and community structure in networks (2015), *Article*
https://www.pnas.org/doi/10.1073/pnas.0601602103#abstract

3 Wikipédia (2022). Méthode de Louvain, *Article*
https://fr.wikipedia.org/wiki/Methode_de_Louvain

4 igraph, *Documentation*
https://igraph.org/python/versions/latest/

5 igraph, *Documentation*
https://igraph.org/python/versions/latest/tutorials/visualize_communities/visualize_communities.html

6 igraph, *Tutoriel*
https://igraph.org/python/api/latest/igraph._igraph.GraphBase.html#Erdos_Renyi